This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

Claim

1. (Currently Amended) A method for managing replicated and migration capable session state for a Java application, comprising the operations of:

executing a Java application on a server, the Java application including multiple entity beans; ~~and~~

executing a replicated state manager, wherein the replicated state manager includes program instructions for managing a replicated and migration capable session state of the Java application using an in-memory database within a Java server process, and wherein the replicated state manager further includes program instructions for replicating ~~the~~ an in-memory state of the Java application to a replicated state server; and

updating and tracking changes for checkpoint mechanism to state objects in response to changes in the state of the application, the replicated state in the replicated state server staying substantially identical to the in-memory state managed by the replicated state manager.

2. (Original) A method as recited in claim 1, wherein the replicated state server is a memory replicated state server.

3. (Original) A method as recited in claim 1, wherein the replicated state server is a disk replicated state server.

4. (Original) A method as recited in claim 1, further comprising the operation of using state replication to different types of state servers to achieve high availability for the Java application.

5. (Original) A method as recited in claim 4, further comprising the operation of recovering state from the state servers during application failure.

6. (Original) A method as recited in claim 5, further comprising the operation of configuring checkpoints using checkpoint policy.

7. (Currently Amended) A method as recited in claim 1, further comprising the operation of storing a state of the entity bean using ~~a~~ the state object.

8. (Original) A method as recited in claim 7, wherein the state object is an in-memory object managed by the replicated state manager with a J2EE server process.

9. (Original) A method as recited in claim 8, further comprising the operation of defining a logical separation between the application and the state objects.

10. (Canceled)

11. (Currently Amended) A system for managing replicated state for a Java platform, comprising:

an application part having an entity bean object;

a managed state part having a ~~first~~ state object, wherein the ~~first~~ state object stores a state of the entity bean object within a memory address space of a J2EE server process; and

a replicated state server that stores a replica of the ~~first~~ state object;

wherein a replicated state in the replicated state server stays substantially identical to an in-memory state managed by a replicated state manager, the replicated state manager tracking updates to the state object in response to changes in the state of the entity bean object.

12. (Original) A system as recited in claim 11, wherein the replicated state server is a memory replicated state server.

13. (Original) A system as recited in claim 11, wherein the replicated state server is a disk replicated state server.

14. (Original) A system as recited in claim 11, wherein a logical separation is defined between the application part and the managed state part.

15. (Canceled)

16. (Currently Amended) A system for managing replicated state for a Java platform, comprising:

an application part having an entity bean object and a related entity bean object;

a managed state part having a first state object and second state object, wherein the first state object stores a state of the entity bean object, and wherein the second state object stores a state of the related entity bean object, and wherein the managed state part maintains a relationship between the entity bean object and the related entity bean object; and

a replicated state server that stores a replica of the first state object and a replica of the second state object, wherein a replicated state in the replicated state server stays substantially identical to an in-memory state managed by a replicated state manager, the replicated state manager tracking updates to the state objects for issuing checkpoints in response to changes in the state of the entity bean object and the related entity bean object.

17. (Original) A system as recited in claim 16, wherein the replicated state server is a memory replicated state server.

18. (Original) A system as recited in claim 16, wherein the replicated state server is a disk replicated state server.

19. (Original) A system as recited in claim 16, wherein a logical separation is defined between the application part and the managed state part.

20. (Canceled)

21. (Currently Amended) A system as recited in claim 16 ~~20~~, wherein the application part maps to a logical schema archive, and wherein the managed state part maps to a physical schema archive.

22. (Original) A system as recited in claim 21, wherein the logical schema archive and the physical schema archive are created in a repository during pre-deployment.

23. (Original) A system as recited in claim 22, wherein the replicated state manager uses the logical schema archive and the physical schema archive at runtime to create the application part and the managed state part with a J2EE server process.

24. (Original) A system as recited in claim 23, wherein an architecture of the replicated state server can be used to plug in a high-performance database server for managing replicated state.

25. (Original) A system as recited in claim 24, wherein the replicated state manager is capable of performing as a transactional resource manager using an in-memory state manager, whereby high throughput is achieved for transactions.

26. (Original) A system as recited in claim 25, wherein the replicated state manager uses a transaction model to manage the checkpoints of the replicated state.

27. (Original) A system as recited in claim 26, the managed state part allows a transaction to proceed when state servers are temporarily unavailable.

28. (Original) A system as recited in claim 16, further including a configurable checkpoint mechanism that is configured using a checkpoint policy.

29. (Original) A system as recited in claim 28, wherein the checkpoint mechanism is configured to be synchronous.

30. (Original) A system as recited in claim 28, wherein the checkpoint mechanism is configured to be asynchronous.

31. (Original) A system as recited in claim 28, wherein the checkpoint mechanism is configured to be boxcarried.

32. (Original) A system as recited in claim 28, wherein the checkpoint mechanism is configured to be single transaction.